

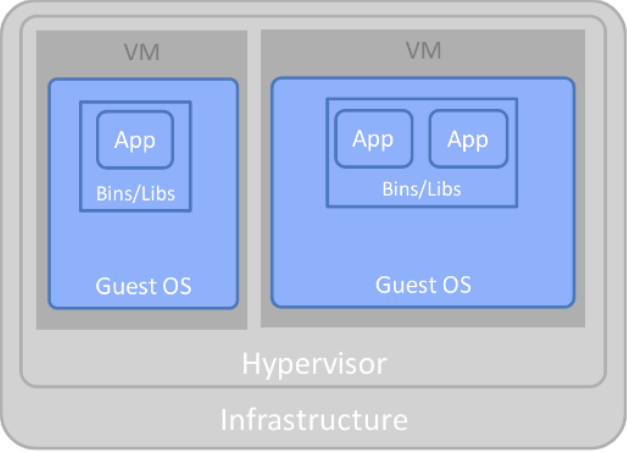
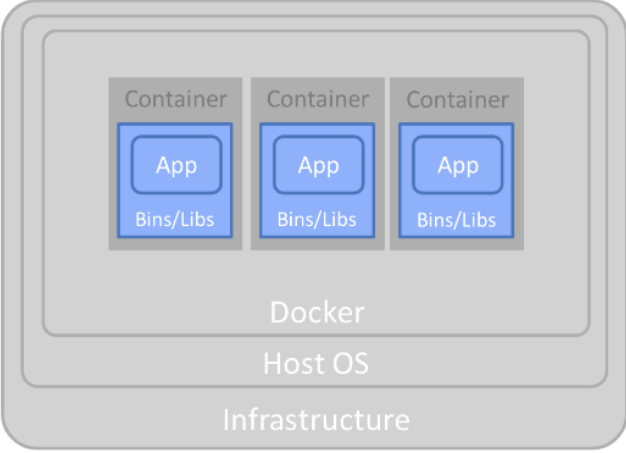
# Exhibit 2

**U.S. Patent No. 7,519,814 (“’814 Patent”)**

Accused Instrumentalities: IBM’s Cloud Kubernetes Service (IKS) and all versions and variations thereof since the issuance of the asserted patent.

**Claim 1**

Claim 1	Accused Instrumentalities
<p>[1pre] 1. In a system having a plurality of servers with operating systems that differ, operating in disparate computing environments, wherein each server includes a processor and an operating system including a kernel a set of associated local system files compatible with the processor, a method of providing at least some of the servers in the system with secure, executable, applications related to a service, wherein the applications are executed in a secure environment, wherein the applications each include an object executable by at least some of the different operating systems for performing a task related to the service, the method comprising:</p>	<p>To the extent the preamble is limiting, IBM practices, through the Accused Instrumentalities, in a system having a plurality of servers with operating systems that differ, operating in disparate computing environments, wherein each server includes a processor and an operating system including a kernel a set of associated local system files compatible with the processor, a method of providing at least some of the servers in the system with secure, executable, applications related to a service, wherein the applications are executed in a secure environment, wherein the applications each include an object executable by at least some of the different operating systems for performing a task related to the service, as claimed.</p> <p><i>See claim limitations below.</i></p> <p><i>See also, e.g.:</i></p> <p>IBM Cloud® Kubernetes Service provides a fully managed container service for Docker (OCI) containers, so clients can deploy containerized apps onto a pool of compute hosts and subsequently manage those containers. Containers are automatically scheduled and placed onto available compute hosts based on your requirements and availability in the cluster.</p> <p><a href="https://www.ibm.com/products/kubernetes-service">https://www.ibm.com/products/kubernetes-service</a></p> <p>With IBM Cloud Kubernetes Service, you can deploy Docker containers into pods that run on your worker nodes. The worker nodes come with a set of add-on pods to help you manage your containers. Install more add-ons through Helm, a Kubernetes package manager. These add-ons can extend your apps with dashboards, logging, IBM Cloud and IBM Watson® services and more.</p> <p><a href="https://www.ibm.com/products/kubernetes-service">https://www.ibm.com/products/kubernetes-service</a></p>

Claim 1	Accused Instrumentalities
	<p data-bbox="646 277 1717 435">Docker is an open source platform that enables developers to build, deploy, run, update and manage <i>containers</i>—standardized, executable components that combine application source code with the operating system (OS) libraries and dependencies required to run that code in any environment.</p> <p data-bbox="636 472 1094 500"><a href="https://www.ibm.com/topics/docker">https://www.ibm.com/topics/docker</a></p> <div data-bbox="646 553 1913 1068"><div><p data-bbox="835 558 1079 586">Virtual Machines</p><p>The diagram illustrates the Virtual Machines (VM) architecture. At the base is the 'Infrastructure' layer. Above it is the 'Hypervisor' layer. The Hypervisor contains two 'VM' (Virtual Machine) boxes. Each VM box contains a 'Guest OS' box. Inside each Guest OS box, there is a 'Bins/Libs' box, and inside that, one or more 'App' boxes. The first VM contains one App, while the second VM contains two Apps.</p></div><div><p data-bbox="1522 558 1682 586">Containers</p><p>The diagram illustrates the Containers architecture. At the base is the 'Infrastructure' layer. Above it is the 'Host OS' layer. The Host OS contains a 'Docker' box. Inside the Docker box, there are three 'Container' boxes. Each Container box contains a 'Bins/Libs' box, and inside that, one 'App' box.</p></div></div> <p data-bbox="636 1101 1598 1128"><a href="https://developer.ibm.com/articles/true-benefits-of-moving-to-containers-1/">https://developer.ibm.com/articles/true-benefits-of-moving-to-containers-1/</a></p>

Claim 1	Accused Instrumentalities
	<p>Containers are executable units of software in which application code is packaged along with its libraries and dependencies, in common ways so that the code can be run anywhere—whether it be on desktop, traditional IT or the cloud.</p> <p>To do this, containers take advantage of a form of operating system (OS) virtualization in which features of the OS kernel (e.g. Linux namespaces and cgroups, Windows silos and job objects) can be leveraged to isolate processes and control the amount of CPU, memory and disk that those processes can access.</p> <p>Containers are small, fast and portable because unlike a virtual machine, containers do not need to include a guest OS in every instance and can instead simply leverage the features and resources of the host OS.</p> <p><a href="https://www.ibm.com/topics/containers">https://www.ibm.com/topics/containers</a></p> <p>With containers, you can isolate the ecosystem to run an application on any host OS (operating system). Containers can wrap code, runtimes, system tools, system libraries—everything that can be installed on a server. Containers are like virtual machines (VMs), but with a key difference in their architectural approach. Images that run on VMs have a full copy of the guest OS, including the necessary binaries and libraries. Images that run on containers share the OS kernel on the host.</p> <p>The Docker Engine builds and spins images on the containers. The engine is a lightweight container runtime that can run on almost any OS. You can run a container anywhere that a Docker Engine can be installed—on bare metal servers, clouds, and even inside a VM. You can move containers from one environment to another without recoding the application.</p> <p>Containers can help DevOps teams in three ways:</p> <ul style="list-style-type: none"> <li>• Increase development productivity by reducing the time spent on environment setup</li> <li>• Eliminate issues that are caused by software dependencies</li> <li>• Avoid inconsistencies when applications are run in different environments</li> </ul> <p>You can use <a href="#">IBM Cloud Kubernetes Service</a> to run containers on IBM Cloud.</p> <p><a href="https://www.ibm.com/garage/method/practices/run/tool_ibm_container/">https://www.ibm.com/garage/method/practices/run/tool_ibm_container/</a>, last accessed on Nov. 17, 2023.</p>

Claim 1	Accused Instrumentalities
	<p>Containers use a form of operating system (OS) virtualization. Put simply, they leverage features of the host operating system to isolate processes and control the processes' access to CPUs, memory and disk space.</p> <p><a href="https://www.ibm.com/blog/containers-vs-vms/">https://www.ibm.com/blog/containers-vs-vms/</a></p> <p>Today Docker containerization also works with Microsoft Windows and Apple MacOS. Developers can run Docker containers on any operating system, and most leading cloud providers, including Amazon Web Services (AWS), Microsoft Azure, and IBM Cloud offer specific services to help developers build, deploy and run applications containerized with Docker.</p> <p><a href="https://www.ibm.com/topics/docker">https://www.ibm.com/topics/docker</a></p>

Claim 1	Accused Instrumentalities
	<p>Containers are often referred to as “lightweight,” meaning they share the machine’s operating system kernel and do not require the overhead of associating an operating system within each application. Containers are inherently smaller in capacity than a VM and require less start-up time, allowing far more containers to run on the same compute capacity as a single VM. This drives higher server efficiencies and, in turn, reduces server and licensing costs.</p> <p>Containers encapsulate an application as a single executable package of software that bundles application code together with all of the related configuration files, libraries, and dependencies required for it to run. Containerized applications are “isolated” in that they do not bundle in a copy of the operating system. Instead, an open source runtime engine (such as the Docker runtime engine) is installed on the host’s operating system and becomes the conduit for containers to share an operating system with other containers on the same computing system.</p> <p><a href="https://www.ibm.com/topics/containerization">https://www.ibm.com/topics/containerization</a></p>

Claim 1	Accused Instrumentalities
	<div data-bbox="640 272 1843 1182"> <p>The diagram illustrates three containers, each labeled 'Container' at the top. Each container is represented as a stack of three colored boxes: an orange box labeled 'App' at the top, a green box labeled 'bins / libs' in the middle, and a yellow box labeled 'OS-specific files' at the bottom. Below each container stack is a grey box labeled 'Base OS/Kernel'. At the very bottom of each column is a blue box labeled 'Hardware'. A central text block below the hardware boxes states: 'Containers share the same base Kernel'. A URL is provided at the bottom: <a href="https://ibm.github.io/kube101/">https://ibm.github.io/kube101/</a></p> </div>
[1a] storing in memory accessible to at least some of the servers a plurality of secure containers of application	The method practiced by IBM through the Accused Instrumentalities includes a step of storing in memory accessible to at least some of the servers a plurality of secure containers of application software, each container comprising one or more of the executable applications and a set of

Claim 1	Accused Instrumentalities
<p>software, each container comprising one or more of the executable applications and a set of associated system files required to execute the one or more applications, for use with a local kernel residing permanently on one of the servers;</p>	<p>associated system files required to execute the one or more applications, for use with a local kernel residing permanently on one of the servers.</p> <p><i>See, e.g.:</i></p> <p>Containers use a form of operating system (OS) virtualization. Put simply, they leverage features of the host operating system to isolate processes and control the processes' access to CPUs, memory and desk space.</p> <p><a href="https://www.ibm.com/blog/containers-vs-vms/">https://www.ibm.com/blog/containers-vs-vms/</a></p> <p>Today Docker containerization also works with Microsoft Windows and Apple MacOS. Developers can run Docker containers on any operating system, and most leading cloud providers, including Amazon Web Services (AWS), Microsoft Azure, and IBM Cloud offer specific services to help developers build, deploy and run applications containerized with Docker.</p> <p><a href="https://www.ibm.com/topics/docker">https://www.ibm.com/topics/docker</a></p>



Claim 1	Accused Instrumentalities
	<div data-bbox="640 272 1843 1182"> <p>The diagram illustrates three containers, each labeled 'Container' at the top. Each container contains three stacked components: 'App' (orange), 'bins / libs' (green), and 'OS-specific files' (yellow). Below the containers is a 'Base OS/Kernel' layer, and at the bottom is a 'Hardware' layer. A text box at the bottom states: 'Containers share the same base Kernel'.</p> <p><a href="https://ibm.github.io/kube101/">https://ibm.github.io/kube101/</a></p> </div>
[1b] wherein the set of associated system files are compatible with a local kernel	In the method practiced by IBM through the Accused Instrumentalities, the set of associated system files are compatible with a local kernel of at least some of the plurality of different operating systems.

Claim 1	Accused Instrumentalities
<p>of at least some of the plurality of different operating systems,</p>	<p><i>See, e.g.:</i></p> <p>Containers are often referred to as “lightweight,” meaning they share the machine’s operating system kernel and do not require the overhead of associating an operating system within each application. Containers are inherently smaller in capacity than a VM and require less start-up time, allowing far more containers to run on the same compute capacity as a single VM. This drives higher server efficiencies and, in turn, reduces server and licensing costs.</p> <p>Containers encapsulate an application as a single executable package of software that bundles application code together with all of the related configuration files, libraries, and dependencies required for it to run. Containerized applications are “isolated” in that they do not bundle in a copy of the operating system. Instead, an open source runtime engine (such as the Docker runtime engine) is installed on the host’s operating system and becomes the conduit for containers to share an operating system with other containers on the same computing system.</p> <p><a href="https://www.ibm.com/topics/containerization">https://www.ibm.com/topics/containerization</a></p>

Claim 1	Accused Instrumentalities
	<div data-bbox="640 272 1843 1182"> <p>The diagram illustrates three containers, each labeled 'Container' at the top. Each container contains three stacked components: 'App' (orange), 'bins / libs' (green), and 'OS-specific files' (yellow). Below the containers is a single 'Base OS/Kernel' layer, and at the bottom is a 'Hardware' layer. A text box below the hardware layer states: 'Containers share the same base Kernel'.</p> <p><a href="https://ibm.github.io/kube101/">https://ibm.github.io/kube101/</a></p> </div>
[1c] the containers of application software excluding a kernel,	<p>In the method practiced by IBM through the Accused Instrumentalities, the containers of application software exclude a kernel.</p> <p><i>See, e.g.:</i></p>

Claim 1	Accused Instrumentalities
	<p>Containers are often referred to as “lightweight,” meaning they share the machine’s operating system kernel and do not require the overhead of associating an operating system within each application. Containers are inherently smaller in capacity than a VM and require less start-up time, allowing far more containers to run on the same compute capacity as a single VM. This drives higher server efficiencies and, in turn, reduces server and licensing costs.</p> <p><a href="https://www.ibm.com/topics/containerization">https://www.ibm.com/topics/containerization</a></p>

Claim 1	Accused Instrumentalities
	<div data-bbox="640 272 1843 1182"> <p style="text-align: center;">Containers share the same base Kernel</p> <p><a href="https://ibm.github.io/kube101/">https://ibm.github.io/kube101/</a></p> </div>
[1d] wherein some or all of the associated system files within a container stored in memory are utilized in place of the	In the method practiced by IBM through the Accused Instrumentalities, some or all of the associated system files within a container stored in memory are utilized in place of the associated local system files that remain resident on the server.

Claim 1	Accused Instrumentalities
<p>associated local system files that remain resident on the server,</p>	<p><i>See, e.g.:</i></p> <p>Rather than spinning up an entire virtual machine, <a href="#">containerization</a> packages together everything needed to run a single application or microservice (along with runtime libraries they need to run). The container includes all the code, its dependencies and even the operating system itself. This enables applications to run almost anywhere — a desktop computer, a traditional IT infrastructure or the cloud.</p> <p>Containers use a form of operating system (OS) virtualization. Put simply, they leverage features of the host operating system to isolate processes and control the processes' access to CPUs, memory and desk space.</p> <p><a href="https://www.ibm.com/blog/containers-vs-vms/">https://www.ibm.com/blog/containers-vs-vms/</a></p>
<p>[1e] wherein said associated system files utilized in place of the associated local system files are copies or modified copies of the associated local system files that remain resident on the server,</p>	<p>In the method practiced by IBM through the Accused Instrumentalities, said associated system files utilized in place of the associated local system files are copies or modified copies of the associated local system files that remain resident on the server.</p> <p><i>See, e.g.:</i></p>

Claim 1	Accused Instrumentalities
	<p>Containerization is the packaging of software code with just the operating system (OS) libraries and dependencies required to run the code to create a single lightweight executable—called a container—that runs consistently on any infrastructure. More portable and resource-efficient than <a href="#">virtual machines (VMs)</a>, containers have become the de facto compute units of modern cloud-native applications.</p> <p>Containerization allows developers to create and deploy applications faster and more securely. With traditional methods, code is developed in a specific computing environment which, when transferred to a new location, often results in bugs and errors. For example, when a developer transfers code from a desktop computer to a VM or from a Linux to a Windows operating system. Containerization eliminates this problem by bundling the application code together with the related configuration files, libraries, and dependencies required for it to run. This single package of software or “container” is abstracted away from the host operating system, and hence, it stands alone and becomes portable—able to run across any platform or cloud, free of issues.</p> <p><a href="https://www.ibm.com/topics/containerization">https://www.ibm.com/topics/containerization</a></p>

Claim 1	Accused Instrumentalities
	<p>With containers, you can isolate the ecosystem to run an application on any host OS (operating system). Containers can wrap code, runtimes, system tools, system libraries—everything that can be installed on a server. Containers are like virtual machines (VMs), but with a key difference in their architectural approach. Images that run on VMs have a full copy of the guest OS, including the necessary binaries and libraries. Images that run on containers share the OS kernel on the host.</p> <p>The Docker Engine builds and spins images on the containers. The engine is a lightweight container runtime that can run on almost any OS. You can run a container anywhere that a Docker Engine can be installed—on bare metal servers, clouds, and even inside a VM. You can move containers from one environment to another without recoding the application.</p> <p>Containers can help DevOps teams in three ways:</p> <ul style="list-style-type: none"> <li>• Increase development productivity by reducing the time spent on environment setup</li> <li>• Eliminate issues that are caused by software dependencies</li> <li>• Avoid inconsistencies when applications are run in different environments</li> </ul> <p>You can use <a href="#">IBM Cloud Kubernetes Service</a> to run containers on IBM Cloud.</p> <p><a href="https://www.ibm.com/garage/method/practices/run/tool_ibm_container/">https://www.ibm.com/garage/method/practices/run/tool_ibm_container/</a>, last accessed on Nov. 17, 2023.</p>
[1f] and wherein the application software cannot be shared between the plurality of secure containers of application software,	<p>In the method practiced by IBM through the Accused Instrumentalities, the application software cannot be shared between the plurality of secure containers of application software.</p> <p><i>See, e.g.:</i></p> <p><a href="#">Containers</a> are made possible by process isolation and virtualization capabilities built into the Linux kernel. These capabilities—such as <i>control groups</i> (Cgroups) for allocating resources among processes, and <i>namespaces</i> for restricting a processes access or visibility into other resources or areas of the system—enable multiple application components to share the resources of a single instance of the host operating system in much the same way that a hypervisor enables multiple <a href="#">virtual machines (VMs)</a> to share the CPU, memory and other resources of a single hardware server.</p> <p><a href="https://www.ibm.com/topics/docker">https://www.ibm.com/topics/docker</a></p>



Claim 1	Accused Instrumentalities
	<p><b>Fault isolation:</b> Each containerized application is isolated and operates independently of others. The failure of one container does not affect the continued operation of any other containers. Development teams can identify and correct any technical issues within one container without any downtime in other containers. Also, the container engine can leverage any OS security isolation techniques—such as SELinux access control—to isolate faults within containers.</p> <p><a href="https://www.ibm.com/topics/containerization">https://www.ibm.com/topics/containerization</a></p>
<p>[1g] and wherein each of the containers has a unique root file system that is different from an operating system's root file system.</p>	<p>In the method practiced by IBM through the Accused Instrumentalities, each of the containers has a unique root file system that is different from an operating system's root file system.</p> <p><i>See, e.g.:</i></p> <p>Limit the number of privileged containers. Containers run as a separate Linux process on the compute host that is isolated from other processes. Although users have root access inside the container, the permissions of this user are limited outside the container to protect other Linux processes, the host file system, and host devices. Some apps require access to the host file system or advanced permissions to run properly. You can run containers in privileged mode to allow the container the same access as the processes running on the compute host. Keep in mind that privileged containers can cause huge damage to the cluster and the underlying compute host if they become compromised. Try to limit the number of containers that run in privileged mode and consider changing the configuration for your app so that the app can run without advanced permissions.</p> <p><a href="https://cloud.ibm.com/docs/containers?topic=containers-security">https://cloud.ibm.com/docs/containers?topic=containers-security</a></p>

Claim 1	Accused Instrumentalities
	<p><a href="#">Containers</a> are made possible by process isolation and virtualization capabilities built into the Linux kernel. These capabilities—such as <i>control groups</i> (Cgroups) for allocating resources among processes, and <i>namespaces</i> for restricting a processes access or visibility into other resources or areas of the system—enable multiple application components to share the resources of a single instance of the host operating system in much the same way that a hypervisor enables multiple <a href="#">virtual machines (VMs)</a> to share the CPU, memory and other resources of a single hardware server.</p> <p><a href="https://www.ibm.com/topics/docker">https://www.ibm.com/topics/docker</a></p>